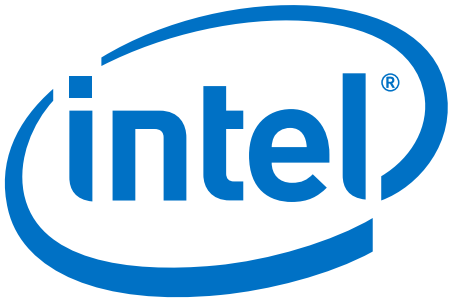

Intel® Stress Bitstreams and Encoder (Intel® SBE) 2017 – HEVC

User Guide

Revision 2.3.0
August 1, 2016

Contents

1 Introduction **1**
1.1 General Overview 1
1.2 Compliance Mode 1
1.3 Extensive testing and parallel execution 2
1.4 HEVC Specific 2
1.5 Example: Random Slices among Tiles 3
1.6 Known Limitations 4

2 Usage of Various Features **5**
2.1 Command Line Options 5
2.2 Parfile Fields and Values 6
2.3 Level Restrictions 6
2.4 Main 10 Profile 7
2.5 Format Range Extension Profiles (RExt) 7
2.6 Error Resilience Encoder 8
2.6.1 Description 8
2.6.2 Important Restrictions 8
2.6.3 Broken Stream Generation 8
2.6.3.1 Bit Randomization 8
2.6.3.2 Packet Randomization 8
2.6.3.3 Syntax Randomization 9
2.7 List of Parameters with the Default Values 10

Legal Information **15**

Revision History

Revision	Date	Description
2.0.1	09/06/2015	Initial import from 2.0 revision with minor fixes
2.1.0	11/11/2015	New features of RExt profiles and error resilience
2.2.0	04/05/2016	Added Monochrome profiles, upgraded split to slices and tiles
2.2.1	06/17/2016	Edited text and legal information
2.3.0	08/01/2016	Added longterm options description

1 Introduction

This document describes the use of Intel® Stress Bitstreams and Encoder
(Intel® SBE) 2016 – HEVC.

1.1 General Overview

Decoder testing is a complex multi-criteria task. Code coverage of 100% lines of a decoder code does not guarantee the decoder is fully compliant with video coding standard. At the same time, creation of millions of streams to test all possible feature combinations is time and storage consuming. Intel® SBE HEVC Encoder partially resolves these two issues. It is a lightweight encoder doing no rate-distortion optimization for mode decision, so it is as fast as the decoder is. Efficient motion estimation and mode decision are not parts of video coding standard and are not required to be tested at the decoder side. So these most time consuming parts are mostly omitted in Intel® SBE HEVC Encoder in favor of speed and flexibility.

All streams generated by random encoder are not needed for validation, it is enough to keep only basic ones. If decoder fails to decode randomly generated stream correctly then it makes sense to extend the test pool with that stream for future regression validation. Intel® SBE HEVC Encoder is a great extension of codec validation in addition to Intel® SBE HEVC Encoder basic bitstreams.

1.2 Compliance Mode

Intel® SBE HEVC Encoder is highly configurable and flexible HEVC syntax encoder tool. Unlike regular encoders, it is not intended to achieve compression but only designed to create a valid specification compliant stream. Compliant streams contain only allowed combinations of syntax elements and their values to test decoder for unusual cases or boundary stress cases where developers usually relax requirements to code development for the sake of higher decoding speed. Decoder must be compatible with any stream so its sloppy optimizations have to be carefully tracked for boundary cases where residuals overflow may break visual representation of the picture.

You can find the recommended validation process with Intel® SBE HEVC Encoder below. It is up to the user to change the flow and to react on pass and fail events and even decide on the criteria of test passing. We recommend to use the latest development branch of reference HM decoder that can be downloaded from https://hevc.hhi.fraunhofer.de/trac/hevc.

As an input, Intel® SBE HEVC Encoder accepts an optional YUV file and a parfile describing testing settings: features to utilize, fixed values, random values. As an output, Intel® SBE HEVC Encoder produces encoded bitstream and optionally writes a YUV file with internal reconstruction data. This file may be used to validate that Intel® SBE HEVC Encoder generated a proper compressed file and that the resulted bitstream is valid.

If there is a mismatch between encoder reconstruct and reference decoder result, you are welcome to report to Software Publisher (your Intel contact) with the case configuration to request the fix, if it fits to your license agreement with Intel. We will always appreciate your feedback.

Intel® SBE HEVC Encoder has the seed parameter (`-s` option) defining initial state of internal random engine. With different seed values one can produce totally different streams with the same scope of randomization defined by parfile. The main purpose of this seed feature is extensive testing with all possible syntax combinations. Besides, this feature can be used to create small bug reproducers by setting frame number to some small value for the parfiles which are known to generate the streams causing failure of the examined decoder.

To summarize, proposed workflow for compliance testing consists of the following steps:

1. Produce test stream by feeding Intel® SBE HEVC Encoder with a parfile and an optional input YUV file.
2. Decode the stream with reference decoder.
3. Verify that Intel® SBE HEVC Encoder reconstruct YUV file matches reference decoder result.
4. Decode the stream with your decoder and verify that its result matches the reference decoder result.
5. Increment the seed parameter and go to the step 1.

1.3 Extensive testing and parallel execution

Intel® SBE HEVC Encoder as fast as ~ 50000 cases a week for a single process. For satisfactory validation, it will need a month execution without fails of a decoder on corner cases. It is possible to execute several processes in parallel on a single computer to increase coverage for the shorter time period as many as hardware memory and CPU cores allows.

1.4 HEVC Specific

To start validation cycle with Intel® SBE HEVC Encoder make a decision regarding your testing agenda and set up certain features and value ranges in input parfile. Parfile is a JSON-formatted file, it contains flags and settings for syntax elements that you can modify. Using the parfile, you can limit the features to a set that is implemented in the decoder under testing and focus on testing these features only.

Intel® SBE HEVC Encoder accepts an optional input YUV file and a parfile and produces one bitstream per call. Therefore the best option for extensive testing is to modify input configuration file outside of random encoder. Figure 1.1 shows how to use many parfiles that may be generated by some script designed by user.

For easier validation, Intel® SBE HEVC Encoder writes hashes of reconstructed pictures into special HEVC suffix SEI messages.

Intel® SBE HEVC Encoder can internally upsample 8-bit YUV source to 10- or 12-bit to test features of relevant HEVC profiles.

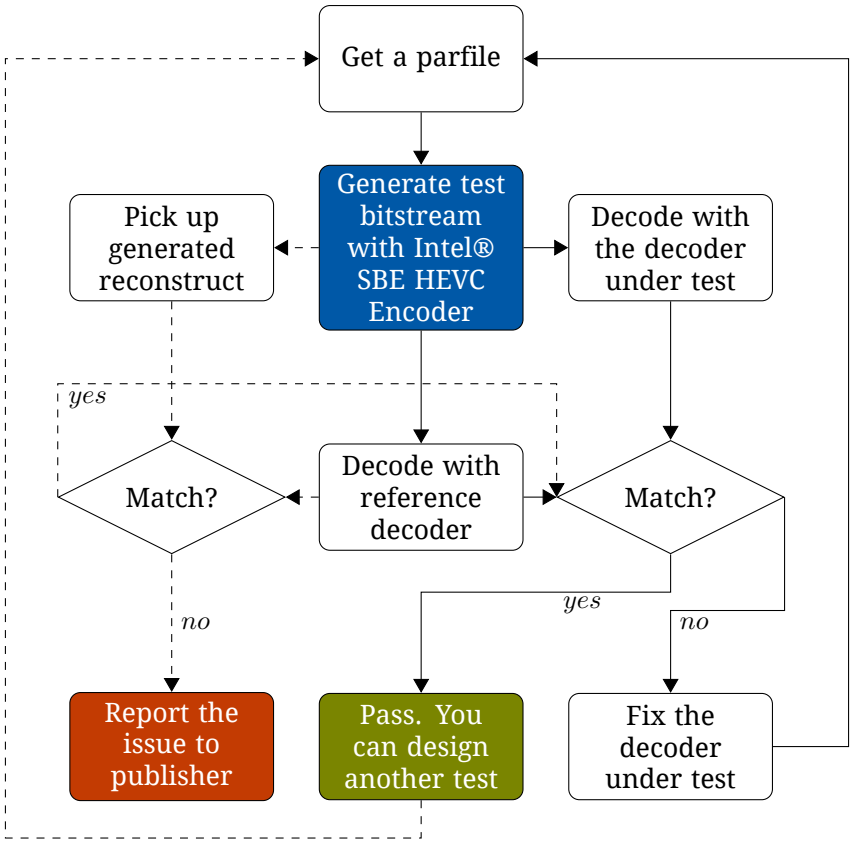

Figure 1.1 – Working with the Intel® SBE HEVC Encoder

1.5 Example: Random Slices among Tiles

Here is a sketch of an algorithm Intel® SBE HEVC Encoder uses to generate a great variety of partitions of a frame to tiles and slices (up to 440 and 600 in Level 6.2, respectively). The algorithm consists of two steps:

1. Suppose Intel® SBE HEVC Encoder decides to divide a frame into T tiles and N slices with slices inside tiles and with tiles inside long slices. Array A of size T is generated at random according to some rules. Encoder specifies the number of slices in a particular tile and the location of the long slices comprising ≥ 1 tiles. At this step, the slice length is measured in number of tiles only. This essentially randomized algorithm takes $O(\max(N, T))$ operations on average.

2. Based on array A, slice lengths are defined in the following way. The length of a long slice is defined as the sum of lengths of tiles in it measured in number of CTBs. Each tile having more than one slice inside it splits into slices using common algorithm of random partitioning of a set into subsets. If the tile consists of M CTBs and there are K slices in this tile with $K \leq M$ then the complexity of that algorithm is $O(K \log K)$ on average. But if for all tiles $K/M \leq \theta < 1$ then the complexity of the algorithm is $O(K)$. Thus the complexity of the second step is only $O(\max(N, T))$.

This algorithm has low complexity and a good coverage of different splits to slices with respect to tile partitioning.

1.6 Known Limitations

The current version of Intel® SBE HEVC Encoder has following limitations:

- Long term references are not covered.
- Different coding structures, i.e. NAL unit types, CPB and DPB features and complicated references are not exhaustively covered by the current randomization.
- Temporal layers are not fully covered by the current randomization though related syntax is touched by randomized algorithm.
- Only several SEI message types may be generated.
- VUI and HRD related syntax is not fully covered.
- Encoder checks not all compatibility constraints.
- Encoder does not give visually pleasant pictures, it has neither RDO nor any kind of rate control.

2

Usage of Various Features

This chapter describes different features of Intel® SBE HEVC Encoder application.

2.1 Command Line Options

Intel® SBE HEVC Encoder is a console application which necessarily requires JSON file (parfile) containing encoding parameters. The name of the parfile should be passed with the option `-p` from the command line.

Some encoding parameters may be controlled by command line options as well. Command line options override matching parfile entries. To view the list of these options from the command line, use `--help`.

2.2 Parfile Fields and Values

Parfile is a JSON file that has the following sections: Stream, VPS, SPS, PPS, Slice, CTB and SEI. Each section has a list of values. Each of them may have one of the following types: string, number, flag, range, or probability `P`. String types are used for file names and tiers. Numbers specify values that do not change during stream generation process. Flags are also unchangeable values which can be `true` or `false` only. Ranges are used for values which vary randomly within specified limits.

Entries of a parfile mostly control the syntax elements of HEVC directly except for some cases when it would be too complicated.

There is no check for bogus entries in JSON-formatted parfile. Intel® SBE HEVC Encoder will silently ignore any of entries that are not listed in section 2.7 of this document.

If parameters in a given parfile are incompatible with HEVC constraints Intel® SBE HEVC Encoder prints a message that it cannot start encoding. Intel® SBE HEVC Encoder may adjust some values at its own discretion if it is still possible to encode bitstream after that. For example, given `num_tile_columns_minus1` to vary in range $[0, 19]$ Intel® SBE HEVC Encoder will adjust the right bound according to the given video resolution, because the minimal picture width for 20 tile columns in HEVC is $256 \times 20 = 5120$. With the `--verbose` option Intel® SBE HEVC Encoder prints messages when encoding parameters are adjusted.

Probability parameters are specified in percents (0–100%) with 0% meaning that the value is always `false`, and 100% meaning that the value is always `true`.

Command line option	Matching parfile entry	Purpose
`--help`	—	Show command line help and exit
`--version`	—	Show version and exit
`--verbose`	`verbose`	Show additional output
`-i <INPUT>`	`source_file`	Set input YUV filename
`-o <OUTPUT>`	`stream_file`	Set output HEVC filename
`-r <RECON>`	`reconstruct_file`	Set reconstruct YUV filename
`--statistics`	`statistics_file`	Print statistics
`-f <FRAMES>`	`num_frames`	Set number of frames to encode
`-s <SEED>`	`seed`	Set seed value for random engine
`--profile <N>`	`profile_idc_range`	Set HEVC profile index
`-p <PARFILE>`	—	Use parfile with encoding parameters
`-l <TIER_LEVEL>`	`tier_level`	Specify and check tier and level
`--ignore_level`	`ignore_level_flag`	Do not check tier and level limits
`--ignore_level_bitrate`	`ignore_level_bitrate_flag`	Do not check level bitrate
`-w <WIDTH>`	`frame_width`	Set frame width
`-h <HEIGHT>`	`frame_height`	Set frame height
`--auto_fix_max_CU_size`	`auto_fix_max_CU_size_flag`	Adjust max CU size for given level
`-bdl <LUMA_BITDEPTH>`	`bit_depth_luma_range`	Set luma bitdepth
`-bdc <CHROMA_BITDEPTH>`	`bit_depth_chroma_range`	Set chroma bitdepth
`--convert_to_10_bit`	`convert_input_to_10_bit_flag`	Convert 8-bit input to 10-bit on-the-fly
`--convert_to_12_bit`	`convert_input_to_12_bit_flag`	Convert 8-bit input to 12-bit on-the-fly
`-b <PROB>`	`random_tu_probability`	Randomize transform coefficients
`-cfmt <CHROMA_FORMAT>`	`chroma_format_idc`	Set HEVC chroma format index

Table 2.1 – Command line options.

2.3 Level Restrictions

HEVC streams have to abide restrictions imposed by tier and level specified in VPS and SPS. By default, random encoder checks whether input parameters satisfy tier and level requirements specified in the parfile. When tier and level are not specified, or keyword `auto` is used, random encoder tries to find the minimum possible tier and level combination necessary for stream encoded with input parameters. With the `--verbose` option it is possible to see in details how tier and level checks are done. To skip tier and level checks one can use the `--ignore_level` command line switch.

Intel® SBE HEVC Encoder has no internal bitrate control, so to satisfy tier and level bitrate and compression requirements one may need to tune some encoding parameters. The following settings significantly affect bitrate:

```
random_tu_probability
ipcm_probability
ipcm_bit_depth_luma_range
ipcm_bit_depth_chroma_range
transquant_bypass_probability
init_qp_minus26
slice_qp_delta
qp_range
```

Generally random encoder produces large streams because unlike

a real encoder it is not trying to achieve any compression over the source picture. With `random_tu_probability` set to 100 pure random signal is generated as a residual regardless of input YUV source. But in most cases, one does not need to produce streams with bitrate limited by tier and level requirement to test a decoder. In such cases, it is possible to use the `--ignore_level_bitrate` option and skip bitrate checks after the bitstream has been generated.

2.4 Main 10 Profile

Main 10 profile allows different bit depths for luma and chroma. To generate streams with varying bit depths it is necessary to set `bit_depth_luma_range` and `bit_depth_chroma_range` parameters to a range of values instead of a fixed value. Bit depth changes can happen only on IDR frame, so it is necessary to set `idr_interval` to a positive value instead of the default zero value. To enable changes for bit depth values in SPS header, it is necessary to enable `sps_header_change_interval`.

Reconstructed YUV file will be in maximal possible bitdepth specified for the stream rounded to bytes. In other words, if `bit_depth_luma_range` is set to $[8, 9]$ and `bit_depth_chroma_range` is set to 8, then both luma and chroma pixels will be packed into 2 byte unsigned shorts.

2.5 Format Range Extension Profiles (RExt)

To generate streams compliant with HEVC Main 4:2:2 10 Format Range Extension profile, user can do one of the following:

- set the `profile_idc_range` input parameter to $[4, 4]$ in the parfile;
- pass the `--profile 4` option from the command line and set the `chroma_format_idc_range` parameter to $[2, 2]$ in the parfile;
- pass the `-cfmt 2` option from the command line.

To enable technologies specific to the Main 4:2:2 10 profile, set the limits required for syntax values using the corresponding input parameters listed below in SPS and/or PPS sections.

Here is an example of a command line with parameters for enabling Main 4:2:2 10 profile:

```
-p 501_main422_10bit.json -i INPUT.yuv -w 1920 -h 1080 -o 1.hevc -f 100 -bdl 10
-bdc 10 -cfmt 2 --profile 4 --convert_to_10_bit
```

where `INPUT.yuv` is a source YUV video sequence, possibly 8-bit.

Main 12, Main 4:4:4 12, Main 4:2:2 12, Main 4:4:4 10 HEVC profiles matching the same HEVC profile index equal to 4 may be enabled by chroma format and bitdepth options of command line. Technologies of the RExt profiles may be enabled by setting entries of parameter file.

Monochrome, Monochrome 12 profiles may be enabled with appropriate chroma format and bitdepth options of command line, profile index equal to 4 and chroma format index set to 0.

2.6 Error Resilience Encoder

2.6.1 Description

Error Resilience Encoder for HEVC is a tool that allows generating broken video streams to test behavior of HEVC decoder with various types of errors. It allows managing type and positioning of errors, that make the bitstream invalid. This tool can be used to pinpoint flaws in error handling and to define the expected behavior on a wide range of possible errors more precisely.

Error Resilience Encoder comes as a separate application and uses JSON parfile that includes the `broken` section.

Error Resilience Encoder is an Intel® SBE HEVC Encoder add-on. Its workflow is as follows:

1. A compliant bitstream is generated with Intel® SBE HEVC Encoder utilizing all of its flexibility.
2. Destructive changes are applied to the generated bitstream, based on the user-defined parameters.

As an input, Error Resilience encoder accepts a YUV file and a parfile, i. e. JSON file, describing testing settings: features to utilize, fixed values, random values. As an output, encoder produces an encoded broken bitstream.

2.6.2 Important Restrictions

Error Resilience Encoder is not assumed to generate valid streams. Currently only first 15 frames of the stream are fully customizable by user. For the rest of the frames if parameters set by user do not meet the required level of degradation, additional errors are embedded into the stream.

2.6.3 Broken Stream Generation

Rules of error generation are described by the `broken` section in a parfile. There are three invalidation options: bitwise randomization, packet-level failures, and corrupted syntax elements. The corresponding subsections in JSON parfile are: `bit`, `packet` and `syntax`.

2.6.3.1 Bit Randomization

The idea of bit randomization is simply to invert some bits in valid stream. The main parameter here is the probability with which a specific bit will be inverted. It is managed by the probability values in subsection `bit`. Each frame of the generated HEVC stream consists of separately coded headers and frame data compressed by CABAC arithmetic coder. Error Resilience Encoder allows to specify separate probabilities for each of these parts. If some of these parameters are not specified explicitly, the default probability is used instead. Bit randomization allows simulating low-level network transmission and storage errors.

2.6.3.2 Packet Randomization

The main idea of the packet randomization is to operate on the packet level, i. e. with sets of bytes. In the `packet` section probability values stand for probabilities for each byte to appear at the beginning

of the corrupted packet. The `size` parameter determines ranged distribution of the length of this packet. This packet could be then cut out from the stream, zeroed, or duplicated, depending on parameter `mode`. As a special case, packet-level randomization allows user to cut out sections or entire frames from bitstream.

2.6.3.3 Syntax Randomization

This section enables damaging of syntax elements whenever it is allowed by bitstream specification. This section consists mostly of the elements of the uncompressed header. Each parameter is a probability with which this feature will be enabled.

2.7 List of Parameters with the Default Values

Parameter	Type	Default	Description
seed	number	1234556789	Pseudorandom generator initial value
num_frames	number	0	Number of frames to generate
start_frame_num	number	0	Ignored
frame_rate	number	30	Stream frame rate
source_file	string	—	Input filename
stream_file	string	—	Output filename
reconstruct_file	string	—	Reconstruct filename
statistics_file	string	—	Statistics filename
verbose	flag	false	Show additional output in console
convert_input_to_10_bit_flag	flag	false	Shifts input by 2 bits to the left
ignore_level_flag	flag	false	Ignore level requirements
ignore_level_bitrate_flag	flag	false	Ignore level bitrate limits
auto_fix_max_CU_size_flag	flag	false	Adjust max CU for given level
disable_sei_flag	flag	false	Disable all SEI messages
vps_header_change_interval	number	1	Create new VPS header every Nth IDR
sps_header_change_interval	number	1	Create new SPS header every Nth IDR
pps_header_change_interval	number	1	Create new PPS header every Nth IDR
idr_interval	number	0	Number of I frames between IDR frames
num_p_range	range	$[0, 0]$	Number of P frames between IDR frames
num_b_range	range	$[0, 0]$	Number of B frames between P frames
max_ref_idx_l0_range	range $[1, 15]$	$[1, 15]$	Max number of frames in L0 list
max_ref_idx_l1_range	range $[1, 15]$	$[1, 15]$	Max number of frames in L1 list
max_b_ref_idx_l0_range	range $[1, 15]$	$[1, 15]$	Max number of L0 frames for B-slices
low_delay_probability	P	50%	Low delay or random access configuration for B-frames
slice_number_range	range $[1, 600]$	$[1, 1]$	Max number of slices in a frame
use_longterm_frames	flag	false	Enables longterm frames generation
max_longterm_frames	number	0	Maximum number of LT frames in generation
longterm_probability	P	0	Probability to encode current frame as longterm

Table 2.2 – Stream section

Parameter	Type	Default	Description
number_of_temporal_layers_range	range	$[1, 1]$	Number of temporal layers
profile_idc_range	range $[1, 2, 4]$	$[1, 1]$	HEVC profile index
tier_level	string	auto	HEVC tier and level

Table 2.3 – VPS section

Parameter	Default	Description
frame_width	1920	Stream width resolution
frame_height	1080	Stream height resolution
chroma_format_idc_range	$[1, 1]$	HEVC chroma format index
num_long_term_ref_pics_sps	$[0, 0]$	HEVC number of longterm reference pic
bit_depth_luma_range	[8, 8]	Bitdepth of luma samples
bit_depth_chroma_range	[8, 8]	Bitdepth of chroma samples
log2_min_cu_size_range	$[3, 6]$	Min CTB size
log2_max_cu_size_range	$[4, 6]$	Max CTB size
log2_min_tu_size_range	$[2, 5]$	Min TU size
log2_max_tu_size_range	$[2, 5]$	Max TU size
log2_max_tu_depth_intra_range	$[0, 4]$	Max transform QT depth for intra CTBs
log2_max_tu_depth_inter_range	$[0, 4]$	Max transform QT depth for inter CTBs
enable_ipcm_probability	50%	Whether to enable IPCM in SPS
log2_min_ipcm_size_range	$[3, 5]$	Min size of CU with IPCM
log2_max_ipcm_size_range	$[3, 5]$	Max size of CU with IPCM
ipcm_bit_depth_luma_range	$[1, 10]$	Bitdepth of luma IPCM samples
ipcm_bit_depth_chroma_range	$[1, 10]$	Bitdepth of chroma IPCM samples
ipcm_loop_filter_disable_probability	50%	Disables loop filter for IPCM mode
enable_temporal_mvp_probability	75%	Enables temporal motion vector predicti
enable_sao_probability	75%	Enables SAO
scaling_list_probability	75%	Enables scaling lists
scaling_list_data_probability	50%	Enables scaling lists data in SPS
amp_enabled_probability	75%	Enables asymmetric motion partitions
strong_intra_smoothing_probability	50%	Tosses intra smoothing
enable_vui_probability	65%	Enables random content in VUI
range_extension_probability	0%	Enables HEVC RExt technologies
transform_skip_rotation_enabled_probability	0%	Enables rotation intra blocks
transform_skip_context_enabled_probability	0%	Particular context for the sig_coeff_f
implicit_rdpcm_enabled_probability	0%	Intra RDPCM
explicit_rdpcm_enabled_probability	0%	Inter RDPCM
intra_smoothing_disabled_probability	0%	Disables intra smoothing
high_precision_offsets_enabled_probability	0%	Enables high precision offsets in WP
persistent_rice_adaptation_enabled_probability	0%	Enables persistent Rice adaptation
cabac_bypass_alignment_enabled_probability	0%	Enables CABAC alignment process

Table 2.4 – SPS section

Parameter	Default	Description
dependent_slice_probability	50%	Enables dependent slices
sign_data_hiding_probability	50%	Enables sign data hiding
cabac_init_present_probability	50%	Tosses cabac_init_presentflag
init_qp_minus26	$[-50, 25]$	Values of QP control in PPS
constrained_intra_probability	50%	Bounds intra prediction to current picture
enable_transform_skip_probability	50%	Enables transform skip
enable_cu_qp_delta_probability	50%	Enables DQP technology
diff_cu_qp_delta_depth_range	$[0, 3]$	Restricts the depth of DQP
enable_transquant_bypass_probability	50%	Tosses transquant_bypass_enabled_flag
weighted_pred_probability	50%	Enables weighted prediction
weighted_bipred_probability	50%	Enables weighted bi-prediction
log2_luma_weight_denom_range	$[0, 7]$	Limits luma denominator
log2_chroma_weight_denom_range	$[0, 7]$	Limits chroma denominator
wavefront_or_tiles_probabilities	$[34, 34]$	Probabilities for WPP, tiles
num_tile_rows_minus1	$[0, 21]$	Number of tile rows
num_tile_columns_minus1	$[0, 19]$	Number of tile columns
uniform_slices_distrib	true	Maintain uniform split to slices
uniform_spacing_probability	50%	Tosses uniform_spacing_flag
loop_filter_across_tiles_probability	50%	Enables loop filtering across tiles
loop_filter_across_slices_probability	50%	Enables loop filtering across slices
deblocking_control_present_probability	75%	Enables deblocking control syntax
deblocking_filter_override_probability	50%	Deblocking control in slice syntax
enable_deblocking_probability	75%	Enables deblocking
scaling_list_data_probability	50%	Enables scaling lists in PPS
log2_parallel_merge_level_range	$[2, 6]$	Limits of Log2ParMrgLevel in HEVC
lists_modification_present_probability	50%	Touches reference lists modification
range_extension_probability	0%	Enables RExt specific syntax in PPS
log2_max_transform_skip_block_size_minus2	[0, 3]	Specifies limit depth for transform skip
chroma_qp_adjustment_enabled_probability	0%	Enables chroma QP offset lists
cu_chroma_qp_adjustment_depth_range	$[0, 3]$	Restricts CU depth for chroma QP offsets
cross_component_prediction_enabled_probability	[0, 3]	Enables CCP

Table 2.5 – PPS section.

Parameter	Default	Description
enable_temporal_mvp_probability	75%	Enables temporal MVP
enable_luma_sao_probability	50%	Enables luma SAO
enable_chroma_sao_probability	50%	Enables chroma SAO
mvd_l1_zero_probability	50%	Tosses mvd_l1_zero_flag
cabac_init_probability	50%	Tosses cabac_init_flag
collocated_from_l0_probability	50%	Tosses collocated_from_l0_flag
collocated_ref_idx_range	$[0, 14]$	Limits to pick out collocated_ref_idx
max_num_merge_cand_range	$[1, 5]$	Limits number of candidates for merge mode
slice_qp_delta	$[-75, 75]$	Limits for QP control in slice
deblocking_filter_override_probability	50%	Deblocking parameters in slice header
enable_deblocking_probability	75%	Enables deblocking in slice
loop_filter_across_slices_probability	50%	Enables loop filter across slices

Table 2.6 – Slice section.

Parameter	Default	Description
random_tu_probability	100%	Probability for every TU to have only random coefficients
ipcm_probability	10%	Probability of intra CU to have PCM mode
transquant_bypass_probability	10%	Probability of transquant bypass CUs
transform_skip_probability	10%	Probability of transform skip CUs
skip_probability	20%	Probability to have SKIP mode in inter PUs
merge_probability	50%	Probability to have MERGE mode in inter PUs
qp_change_probability	25%	Probability to change QP with every CU
qp_range	$[0, 51]$	Limits to pick out QP for CU
intra_in_inter_probability	10%	Probability of generating intra prediction inside P and B slices
intra_NxN_probability	50%	Probability of INTRA_NxN vs INTRA_2Nx2N

Table 2.7 – CTB section.

Parameter	Default	Description
activate_parameter_sets_probability	20%	Active parameter sets
buffering_period_probability	20%	Buffering period
picture_timing_probability	20%	Picture timing
decoded_picture_hash_probability	100%	Decoded picture hash
filler_probability	80%	Filler payload
filler_payload_length_range	$[1, 50]$	Number of payload bytes for filler SEI
user_data_registered_probability	80%	User data registered by Rec. ITU-T T.35
user_data_registered_length_range	$[1, 50]$	Number of payload bytes for user data registered SEI
user_data_unregistered_probability	80%	User data unregistered
user_data_unregistered_length_range	$[1, 50]$	Number of payload bytes for user data unregistered SEI

Table 2.8 – SEI section.

Parameter	Default	Description
slice_header_broken_probability	0.00001	Affects slice header
code_ptl_broken_probability	0.00001	Affects Profile-Tier-Level header
scaling_list_broken_probability	0.00001	Damages scaling lists
vps_broken_probability	0.00001	Damages VPS
sps_broken_probability	0.00001	Damages SPS
pps_broken_probability	0.00001	Damages PPS
seq_broken_probability	0.00001	Damages sequence
nal_broken_probability	0.00001	Damages NAL units
frame_data_probability	0.00001	Damages frame
xcode_scaling_list_probability	0.00001	Damages scaling lists
short_term_ref_pic_set_probability	0.00001	Damages short term refs
hrd_parameters_probability	0.0001	Damages HRD

Table 2.9 – Bit section.

Parameter	Default	Description
slice_header_broken_probability	0.00001	Affects slice header
code_ptl_broken_probability	0.00001	Affects Profile-Tier-Level header
scaling_list_broken_probability	0.00001	Damages scaling lists
vps_broken_probability	0.00001	Damages VPS
sps_broken_probability	0.00001	Damages SPS
pps_broken_probability	0.00001	Damages PPS
seq_broken_probability	0.00001	Damages sequence
code_profile_tier_broken_probability	0.00001	Affects Profile-Tier-Level header
nal_broken_probability	0.00001	Damages NAL units
frame_data_probability	0.00001	Damages frame
xcode_scaling_list_probability	0.00001	Damages scaling lists
short_term_ref_pic_set_probability	0.00001	Damages short term refs
hrd_parameters_probability	0.000001	Damages HRD
size	$[2, 8]$	Limits of variadic corrupted chunk
mode	$[1, 0, 0]$	Loss, duplicate or zero

Table 2.10 – Packet section.

Parameter	Default	Description
preset_syntax_probability	0.01	Enable broken for all headers with x probability
sps_preset_probability	0.01	Enable broken for SPS header
rps_preset_probability	0.001	Enable broken for RPS
vps_preset_probability	0.01	Enable broken for VPS header
pps_preset_probability	0.001	Enable broken for PPS header
slice_preset_probability	0.001	Enable broken for Slice header
scaling_list_preset_probability	0.001	Enable broken for Scaling List
x_scaling_list_preset_probability	0.001	Enable broken for x scaling list
profile_tier_preset_probability	0.01	Enable broken for profile tier

Table 2.11 – Broken syntax section.

Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

HEVC (H.265), MPEG-1, MPEG-2, MPEG-4, H.261, H.263, AVC (H.264), MP3, DV, VC-1, MJPEG, AC3, AAC, G.711, G.722, G.722.1, G.722.2, AMRWB, Extended AMRWB (AMRWB+), G.167, G.168, G.169, G.723.1, G.726, G.728, G.729, G.729.1, GSM AMR, GSM FR are international standards promoted by ISO, IEC, ITU, ETSI, 3GPP and other organizations. Implementations of these standards, or the standard enabled platforms may require licenses from various entities, including Intel Corporation.

Intel, the Intel logo, Intel Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel.

Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804